# Data Movement Engine – Part 2 – General Configuration

**Purpose:**

The main purpose of this engine is to move data from one source to another source with basic data validation checks between source and target.

**Checks included between source and target:**

* Total records count
* Distinct records count
* Columns count
* Null Records count

**Different Source systems used:**

* S3 bucket
* MYSQL Database
* Redshift Database
* Standalone/cluster Hadoop HDFS File system
* Standalone/cluster Hadoop Hive database
* Remote HDFS
* Remote Hive database

NOTE: More sources can be added if needed

**Configuration details:**

Below are the configuration details we need to use while running the spark job for multiple source systems.

**S3 as source:**

Create a S3 bucket on AWS and note down the access and secret key to access the bucket.

Place your file in the bucket.

<https://aws.amazon.com/blogs/security/wheres-my-secret-access-key/> -/Reference

Use the below parameters to pull a **JSON** file/files from S3 bucket in the parameter file while running the job.

**Note: I have used sourceFileName, sourceFileFormat parameters to read S3 bucket and HDFS files.**

**If we need to read a single file then we can mention the file name as below.**

**Ex: file name is movie.json**

**sourceFileName becomes movie**

**and sourceFileFormat becomes json**

**If we want to read multiple files with extension json then we**

**sourceFileName becomes \***

**and sourceFileFormat becomes json**

|  |
| --- |
| {  "sourceType": "S3",  "sourceAccessKey": "",  "sourceSecretKey": "",  "sourceBucketPath": "",  "sourceFileName": "",  "sourceFileFormat": "",  "sourceMultiLineFlag": "<true/false for multi-line json file>" //optional  } |

Reading files with a **delimiter**:

|  |
| --- |
| {  "sourceType": "S3",  "sourceAccessKey": "",  "sourceSecretKey": "",  sourceHeader": "<true/false in case your file has a header>" //optional,  "sourceFileName": "",  "sourceFileFormat": "",  "sourceDelimiter": "<fields delimiter ex: , or | etc>" //optional : reads csv files by default  } |

**S3 as Target:** writing data to S3 bucket path

|  |
| --- |
| {  "targetType": "S3",  "targetAccessKey": "",  "targetSecretKey": "",  "targetBucketPath": "",  "targetSaveMode": ""  } |

**MYSQL as Source:**

Reading Mysql table data:

|  |
| --- |
| {  "sourceType": "MYSQL",  "sourceIP": "",  "sourcePort": "",  "sourceSchemaName": "",  "sourceTableName": "",  "sourceUsername": "",  "sourcePassword":""  } |

Read data using a query:

|  |
| --- |
| {  "sourceType": "MYSQL",  "sourceIP": "",  "sourcePort": "",  "sourceSchemaName": "",  "sourceQuery": "< your query>",  "sourceUsername": "",  "sourcePassword":"" } |

**MYSQL as Target:** write data to MYSQL Table

|  |
| --- |
| {  "targetType": "MYSQL",  "targetIP":"",  "targetPort":"",  "targetSchemaName":"",  "targetTableName":"",  "targetUsername":"",  "targetPassword":"",  "targetSaveMode":""  } |

**HDFS as Source:**

Reading JSON data from remote public HDFS location:

|  |
| --- |
| {  "sourceType": "REMOTEHDFS",  "sourceIP":"",  "sourcePort":"",  "sourcePath":"< hdfs files path>",  "sourceFileFormat":"< hdfs files extension>" ,  "sourceFileName":""  } |

Reading files with a delimiter:

|  |
| --- |
| {  "sourceType": "REMOTEHDFS",  "sourceIP":"",  "sourcePort":"",  "sourcePath":"< hdfs files path>",  "sourceFileName":"",  "sourceFileFormat":"< hdfs files extension>" ,  "sourceHeader":"<true/false in case of files have headers>",  "sourceDelimiter":"",  "sourceFileInputDateFormat":"" //optional -> input file date field format  } |

Remote HDFS as Target:

|  |
| --- |
| {  "targetType": "REMOTEHDFS",  "targetIP":"",  "targetPort":"",  "targetPath":""  } |

Reading local HDFS JSON File in a standalone or cluster hadoop environment:

|  |
| --- |
| {  "sourceType": "HDFS",  "sourcePath":"< hdfs files path>",  "sourceFileFormat":"< hdfs files extension>",  "sourceFileName":""  } |

Reading local HDFS File with a delimiter:

|  |
| --- |
| {  "sourceType": "HDFS",  "sourcePath":"< hdfs files path>",  "sourceFileFormat":"< hdfs files extension>" ,  "sourceFileName":"",  "sourceHeader":"<true/false in case of files have headers>",  "sourceDelimiter":"" ,  "sourceFileInputDateFormat":"" //optional -> input file date field format  } |

Writing data to HDFS in a standalone or cluster hadoop environment:

|  |
| --- |
| {  "targetType": "HDFS",  "targetPath":""  } |

**Hive as source:**

Reading Remote Hive table Data:

|  |
| --- |
| {  "sourceType": "REMOTEHIVE",  "sourceIP":"",  "sourcePort":"",  "sourceSchemaName":"",  "sourceTableName":"",  "sourceColumns":"" //optional . we can used this when we want to read particular columns  } |

Reading Remote Hive data by passing a query:

|  |
| --- |
| {  "sourceType": "REMOTEHIVE",  "sourceIP":"",  "sourcePort":"",  "sourceSchemaName":"",  "sourceQuery":""  } |

Write Data to remote Hive Table:

|  |
| --- |
| {  "targetType": "REMOTEHIVE",  "targetIP":"",  "targetPort":"",  "targetSchemaName":"",  "targetTableName":"",  "targetSaveMode":""  } |

Reading Hive in a standalone or cluster Hadoop Environment:

|  |
| --- |
| {  "sourceType": "HIVE",  "sourceSchemaName":"",  "sourceTableName":"",  "sourceColumns":"" //optional . we can used this when we want to read particular columns  } |

Reading local hive using a query:

|  |
| --- |
| {  "sourceType": "HIVE",  "sourceSchemaName":"",  "sourceQuery":""  } |

**Redshift as source:**

Reading data from Redshift table:

<https://docs.databricks.com/data/data-sources/aws/amazon-redshift.html>

|  |
| --- |
| {  "sourceType": "REDSHFIT",  "sourceURL":"jdbc:redshift://redshifthost:5439/database?user=username&password=pass",  "sourceTableName":"my\_table",  "sourceTempDir":"", //S3 bucket temp dir path for redshift  "sourceAccessKey": "",  "sourceSecretKey": ""  } |

Read Data from a query:

|  |
| --- |
| {  "sourceType": "REDSHIFT",  "sourceURL":"jdbc:redshift://redshifthost:5439/database?user=username&password=pass",  "sourceQuery":"select \* from my\_table",  "sourceTempDir":"", //S3 bucket temp dir path for redshift  "sourceAccessKey": "",  "sourceSecretKey": ""  } |

Write data to Redshift table:

|  |
| --- |
| {  "targetType": "REDSHIFT",  "targetURL":"jdbc:redshift://redshifthost:5439/database?user=username&password=pass",  "targetTableName":"my\_table",  "targetTempDir":"", //S3 bucket temp dir path for redshift  "targetAccessKey": "",  "targetSecretKey": "",  "targetSaveMode":""  } |

Write data to Redshift table using IAM Role Based Authentication:

|  |
| --- |
| {  "targetType": "REDSHIFT",  "targetURL":"jdbc:redshift://redshifthost:5439/database?user=username&password=pass",  "targetTableName":"my\_table",  "targetTempDir":"", //S3 bucket temp dir path for redshift  "targetAccessKey": "",  "targetSecretKey": "",  "targetSaveMode":"",  "targetAWSIAMRole":""  } |

**JDBC Connection String as source:**

This function can be used when we are reading data from a source using JDBC Connection. We path the DB credentials in a param file with a respective jar file while running the job.

Read from a table:

|  |
| --- |
| {  "sourceType": "JDBCURL",  "sourceJDBCUrl": " jdbc:redshift://redshifthost:5439/database ",  "sourceDriverName": "",  "sourceTableName": "",  "sourceUsername": "",  "sourcePassword":""  } |

Read using a Query:

|  |
| --- |
| {  "sourceType": "JDBCURL",  "sourceJDBCUrl": " jdbc:redshift://redshifthost:5439/database",  "sourceDriverName": "",  "sourceQuery": "",  "sourceUsername": "",  "sourcePassword":""  } |

Write to database:

|  |
| --- |
| {  "targetType":"JDBCURL"  "targetJDBCUrl":"jdbc:redshift://redshifthost:5439/database",  "targetDriverName":"com.mysql.jdbc.Driver",  "targetTempDir":"", //S3 bucket temp dir path for redshift  "targetTableName": "",  "targetUsername": "",  "targetSaveMode":"",  "targetPassword"":""  } |

Note: Please refer the code if you face any issues. This framework has not been tested for all the sources mentioned above.